
ELSEVIER

Journal of Computational and Applied Mathematics 133 (2001) 545–554

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

www.elsevier.com/locate/cam

Locating and computing in parallel all the simple roots of special functions using PVM

V.P. Plagianakos *, N.K. Nousis, M.N. Vrahatis

Department of Mathematics, University of Patras, U.P. Artificial Intelligence Research Center-UPAIRC, GR-26500 Patras, Greece

Received 2 November 1999; received in revised form 16 June 2000

Abstract

An algorithm is proposed for locating and computing in parallel and with certainty all the simple roots of any twice continuously differentiable function in any specific interval. To compute with certainty all the roots, the proposed method is heavily based on the knowledge of the total number of roots within the given interval. To obtain this information we use results from topological degree theory and, in particular, the Kronecker–Picard approach. This theory gives a formula for the computation of the total number of roots of a system of equations within a given region, which can be computed in parallel. With this tool in hand, we construct a parallel procedure for the localization and isolation of all the roots by dividing the given region successively and applying the above formula to these subregions until the final domains contain at the most one root. The subregions with no roots are discarded, while for the rest a modification of the well-known bisection method is employed for the computation of the contained root. The new aspect of the present contribution is that the computation of the total number of zeros using the Kronecker–Picard integral as well as the localization and computation of all the roots is performed in parallel using the parallel virtual machine (PVM). PVM is an integrated set of software tools and libraries that emulates a general-purpose, flexible, heterogeneous concurrent computing framework on interconnected computers of varied architectures. The proposed algorithm has large granularity and low synchronization, and is robust. It has been implemented and tested and our experience is that it can massively compute with certainty all the roots in a certain interval. Performance information from massive computations related to a recently proposed conjecture due to Elbert (this issue, J. Comput. Appl. Math. 133 (2001) 65–83) is reported.

MSC: 33C10; 68Q22; 65H05; 65D20; 47H11; 55M25; 65Y05; 65Y10; 68M10

Keywords: Elbert's conjecture; Special functions; Bessel functions; Computation of special functions; Construction of tables of special functions; Parallel and distributed algorithms; Parallel virtual machine; Zero isolation; Kronecker–Picard theory; Topological degree theory; Computing simple roots; Bisection method

* Corresponding author.

E-mail address: vpp@math.upatras.gr (V.P. Plagianakos).

0377-0427/01/$ - see front matter
PII: S0377-0427(00)00675-0

1. Introduction

Many problems in different areas of science, such as mechanics, physical sciences, statistics, operation research, etc., are reduced to the problem of finding all the roots or the extrema of a function $f: [a,b] \subset \mathbb{R} \to \mathbb{R}$, in a given interval. The importance of the problem has attracted the attention of many research efforts and as a result many different approaches to the problem exist. Regarding the special functions, Lozier and Olver [8,10] have provided a survey of algorithms and software for the numerical evaluation of special functions. They have pointed out that the currently available software for special functions (including the zerofinding approach) exhibits gaps and defects in comparison to the needs of modern high-performance computing and also, surprisingly, in comparison to what could be constructed from current algorithms (see for example [14,21]). In particular the software regarding the zeros of Bessel functions has a low cumulative score in [8, p. 351] and thus the computation of the zeros of special functions is an area of particular need.

On the other hand, parallel processing, i.e., the method of having many small tasks solve one large problem, has emerged as a key enabling technology in modern computing [9]. The past several years have witnessed an ever-increasing acceptance and adoption of parallel processing, both for high-performance scientific computing and for more "general-purpose" applications, as a result of the demand for higher performance, lower cost, and sustained productivity. The acceptance has been facilitated by two major developments: massive parallel processors (MPPs), and the widespread use of distributed computing.

The most important factor in distributed computing is the high cost of the hardware. Large MPPs typically cost more than $10 million. In contrast, users see very little cost in running their problems on a local set of existing computers. The parallel virtual machine (PVM) is a de facto standard message passing interface. It is an integrated set of software tools and libraries that emulates a general-purpose, flexible, heterogeneous concurrent computing framework on interconnected computers of varied architectures. PVM is designed to link computing resources and provide users with a parallel platform for running their computer applications, irrespective of the number of different computer architectures they use and where those computers are located. It is capable of harnessing the combined resources of typically heterogeneous networked computing platforms to deliver high levels of performance and functionality. Its key concept is that it makes a collection of computers to appear as one large *virtual* machine, hence its name [3].

In this paper we propose an algorithm for locating and computing *in parallel* and with *certainty* all the simple roots of any twice continuously differentiable function in any given interval. The rest of the paper is organized as follows: In Section 2, we briefly review the notion of topological degree and the Picard and Kronecker approach. In Section 3, we apply the results of the previous section to twice continuously differentiable functions of one variable and in the next section we present the modified bisection method. In Section 5 we propose an algorithm to locate and compute all the simple roots in parallel. Experiments and simulation results are presented in Section 6. Finally, the paper ends with some concluding remarks and a short discussion for further work.

2. The topological degree for the localization of zeros

To obtain the total number of roots within any predetermined interval we use results from topological degree theory and in particular the *Kronecker–Picard* approach [12,13]. This theory gives a

formula for the computation of the total number of roots of a system of equations within a given region.

To define the topological degree we suppose that the solutions of the equation:

$$F_n(x) = \mathcal{O}_n \equiv (0, 0, \ldots, 0), \tag{2.1}$$

where $F_n = (f_1, \ldots, f_n)$: $\bar{\mathscr{D}}^n \subset \mathbb{R}^n \to \mathbb{R}^n$ is twice continuously differentiable in the open and bounded domain $\mathscr{D}^n$, are not located on the boundary $b(\mathscr{D}^n)$ of $\mathscr{D}^n$ and they are simple, i.e., the Jacobian determinant of F_n at these roots is non-zero.

The *topological degree of* F_n *at* $\mathcal{O}_n$ *relative to* $\mathscr{D}^n$ is denoted by $\deg[F_n, \mathscr{D}^n, \mathcal{O}_n]$ and can be defined by the following sum:

$$\deg[F_n, \mathscr{D}^n, \mathcal{O}_n] = \sum_{x \in F_n^{-1}(\mathcal{O}_n)} \operatorname{sgn} J_{F_n}(x), \tag{2.2}$$

where J_{F_n} indicates the determinant of the Jacobian matrix and sgn defines the well-known three-valued sign function. The notion of topological degree can be generalized when F_n is only continuous [1,11]. Kronecker's theorem states that Eq. (2.1) has at least one root in $\mathscr{D}^n$ if $\deg[F_n, \mathscr{D}^n, \mathcal{O}_n] \neq 0$.

The definition of the topological degree actually indicates that its value is equal to the number of simple solutions of Eq. (2.1) for which the determinant of the Jacobian matrix is positive, minus the number of simple roots for which the Jacobian determinant is negative. Evidently, if all of them give the same Jacobian determinant sign, then the total number $\mathscr{N}^r$ of simple roots of $F_n(x)$ can be obtained by the value of $\deg[F_n, \mathscr{D}^n, \mathcal{O}_n]$. To retain the same sign of the Jacobian determinant, Picard has considered the following extensions of the function F_n and the domain $\mathscr{D}^n$:

$$F_{n+1} = (f_1, \ldots, f_n, f_{n+1}) : \mathscr{D}^{n+1} \subset \mathbb{R}^{n+1} \to \mathbb{R}^{n+1}, \tag{2.3}$$

where $f_{n+1} = y J_{F_n}$ and $\mathscr{D}^{n+1}$ is the direct product of the domain $\mathscr{D}^n$ with an arbitrary interval of the real y-axis containing the point $y = 0$. Then the following system of equations:

$$\begin{aligned} &f_i(x_1, x_2, \ldots, x_n) = 0, \quad i = 1, \ldots, n, \\ &y J_{F_n}(x_1, x_2, \ldots, x_n) = 0, \end{aligned} \tag{2.4}$$

possesses the same simple roots with $F_n(x)$, provided $y = 0$. Also, it is easily seen that the Jacobian determinant of (2.4) is equal to $J_{F_n}^2$ which is always positive. So, we conclude that the total number $\mathscr{N}^r$ of solutions of Eq. (2.1) is

$$\mathscr{N}^r = \deg[F_{n+1}, \mathscr{D}^{n+1}, \mathcal{O}_{n+1}]. \tag{2.5}$$

Thus, the total number of zeros can be obtained by the value of the topological degree. The topological degree can be computed by the Kronecker integral:

$$\deg[F_n, \mathscr{D}^n, \mathcal{O}_n] = \frac{1}{\Omega_n} \int \int_{b(\mathscr{D}^n)} \cdots \int \frac{\sum_{i=1}^{n} A_i \, \mathrm{d}x_1 \cdots \mathrm{d}x_{i-1} \, \mathrm{d}x_{i+1} \cdots \mathrm{d}x_n}{(f_1^2 + f_2^2 + \cdots + f_n^2)^{n/2}}, \tag{2.6}$$

where A_i define the following determinants:

$$A_i = (-1)^{n(i-1)} | \, F_n \; \partial_1 F_n \; \cdots \; \partial_{i-1} F_n \; \partial_{i+1} F_n \; \cdots \; \partial_n F_n \, |,$$

while $\partial_j F_n$ defines the column vector $(\partial f_1 / \partial x_j, \ldots, \partial f_n / \partial x_j)$ and Ω_n denotes the surface of a hypersphere in $\mathbb{R}^n$ with radius unity, i.e., $\Omega_n = 2\pi^{n/2} / \Gamma(n/2)$.

3. Computing the number of simple roots

To study the real zeros of special functions, we focus on the problem of calculating the total number of simple roots of a real twice continuously differentiable function $f(x)$, defined in a predetermined interval $[a,b]$, where a and b are arbitrarily chosen so that $f(a)f(b)\neq 0$. According to Picard's extension we consider the function $F_2=(f_1,f_2)$: $\mathscr{P}^2\subset\mathbb{R}^2\to\mathbb{R}^2$ and the corresponding system:

$$f_1(x,y)=f(x)=0,\quad f_2(x,y)=yf'(x)=0, \tag{3.1}$$

where the prime denotes differentiation and $\mathscr{P}^2$ is the rectangular parallelepiped $[a,b]\times[-\xi,\xi]$ in the (x,y)-plane with ξ an arbitrary positive constant. Since the roots are simple, which means $f'(x)\neq 0$ for $x\in f^{-1}(0)$, it is easily seen that the solutions of system (3.1) in $\mathscr{P}^2$ and these of $f(x)=0$ in (a,b) are the same. Also, since $J_{F_2}=f'^2$, the total number of simple zeros $\mathscr{N}^r$ of $f(x)$ in (a,b) are given by

$$\mathscr{N}^r=\deg[F_2,\mathscr{P}^2,\mathscr{O}_2]. \tag{3.2}$$

For the computation of the topological degree of F_2 we apply Kronecker integral (2.6) for $n=2$. Using the relations $\mathrm{d}f_j=(\partial f_j/\partial x_1)\,\mathrm{d}x_1+(\partial f_j/\partial x_2)\,\mathrm{d}x_2$, $j=1,2$ we obtain

$$\mathscr{N}^r=\frac{1}{2\pi}\oint_{b(\mathscr{P}^2)}\frac{f_1\,\mathrm{d}f_2-f_2\,\mathrm{d}f_1}{f_1^2+f_2^2}=2\frac{1}{\pi}\oint_{b(\mathscr{P}^2)}\mathrm{d}\arctan\left(\frac{f_2}{f_1}\right). \tag{3.3}$$

Replacing f_1 and f_2 by virtue of (3.1) and performing the integration in (3.3) we finally get

$$\mathscr{N}^r=-\frac{1}{\pi}\left[\xi\int_a^b\frac{f(x)f''(x)-f'^2(x)}{f^2(x)+\xi^2f'^2(x)}\,\mathrm{d}x-\arctan\left(\frac{\xi f'(b)}{f(b)}\right)+\arctan\left(\frac{\xi f'(a)}{f(a)}\right)\right]. \tag{3.4}$$

Note that it has been explicitly shown in [12,13] that relation (3.4) is independent of the value of ξ.

4. The modified bisection method

Having isolated one root of the function within an interval, we can use a modified version of the bisection method to compute it [17,18] which is based on the nonzero value of the topological degree (or alternatively Bolzano's criterion) [19]. To compute a solution of $f(x)=0$ where $f:[a,b]\subset\mathbb{R}\to\mathbb{R}$ is continuous, the following iterative formula can be used:

$$x_{i+1}=x_i+c\,\operatorname{sgn}f(x_i)/2^{i+1},\quad i=0,1,\ldots \tag{4.1}$$

with $x_0=a$ and $c=\operatorname{sgn}f(a)(b-a)$. Iterations (4.1) converge to a root $r\in(a,b)$ if for some x_i, $i=1,2,\ldots$, the following holds:

$$\operatorname{sgn}f(x_0)\operatorname{sgn}f(x_i)=-1.$$

The number of iterations η, which are required to obtain an approximate root r^* such that $|r-r^*|\leqslant\varepsilon$ for some $\varepsilon\in(0,1)$, is given by

$$\eta=\lceil\log_2((b-a)\varepsilon^{-1})\rceil, \tag{4.2}$$

where the notation $\lceil\cdot\rceil$ refers to the smallest integer not less than the real number quoted.

Alternatively, any one-dimensional rootfinding method can be used. The reason for choosing the bisection method is that it always converges within the given interval and it is a globally convergent method. Moreover, it has a great advantage since it is optimal, i.e., it possesses asymptotically the best-possible rate of convergence [15]. Also, using relation (4.2) it is easy to have beforehand the number of iterations that are required for the attainment of an approximate root to a predetermined accuracy. Furthermore, it requires only the algebraic signs (one bit of information) of the function values to be computed, as it is evident from (4.1), thus it can be applied to problems with imprecise function values. As a consequence for problems where the function value follows as a result of an infinite series (e.g., Bessel or Airy functions) it can be shown [20,22,23] that the sign stabilizes after a relatively small number of terms of the series and the calculations can be sped up considerably. Finally, the bisection method can be parallelized easily and effectively.

5. The algorithms

With these tools in hand, one can construct a parallel procedure for the isolation and computation of all the simple roots using a divide-and-conquer technique, i.e., by dividing the given region successively and applying the above formula to these subregions until the final domains contain at most one root. The subregions with no roots are discarded, while for the rest the modified bisection method is employed for the computation of the contained root for the reasons explained previously. To isolate and compute all the simple roots using PVM, we have used the "master–slave" computational model. We have used 10 Beowulf-style [16] slave nodes and one master.

5.1. The algorithm of the master

In the beginning, the master adds the slaves to the PVM, subdivides the initial interval $[a,b]$ into subintervals, one for each slave, and sends them to the slaves. While there are slaves that work, the master receives an interval and the corresponding number of roots. If the number of roots is equal to one, then it stores the root, otherwise it "pushes" the interval and the number of roots to a "stack". Now, while there are idle slaves and the stack is not empty, it "pops" an interval from the stack and sends it to an idle slave. When all the roots are found, the master sends termination signals to all the slaves and shuts the PVM down.

```
procedure master
initializeAllSlaves
h: = (b-a) / numOfSlaves
for slave : = 1 to numOfSlaves do
            SendInterval(slave, a, a + h, 0)
            a: = a+h
while numOfIdleSlaves < > numOfSlaves do
            RcvInterval(slave, a, b, numOfRoots, posRoot)
            if numOfRoots =1 then print(posRoot, f(posRoot))
            if numOfRoots > 1 then StackPush(a, b, numOfRoots)
```

```
        while (numOfIdleSlaves > 0) and not EmptyStack do
               StackPop(a, b, numOfRoots)
               SendIterval(FindIdleSlave(), a, b, numOfRoots)
ShutdownAllSlaves
```

5.2. The algorithm of the slaves

Each slave receives an interval (a,b), and computes the midpoint, *midPoint*, of the interval. If the slave has also received the number of roots in (a,b), it computes the number of roots of the interval $(a, midPoint)$ using (3.4) and calculates the number of roots in $(midPoint, b)$, with a simple subtraction. Otherwise it uses (3.4) for both intervals.

If an interval has only one root, the slave finds it using the modified bisection method. The same procedure is repeated for the $(midPoint, b)$ interval. Finally, it returns to the master the new intervals with the corresponding number of roots. A high-level description of the algorithm is given below:

```
procedure slave
start:
  RcvInterval(a, b, numOfRoots)
  midPoint : = (a+b)/2
  numOfLeftRoots := FindRoots(a, midPoint)
  numOfRightRoots := numOfRoots - numOfLeftRoots
  if numOfRoots =0 then numOfRightRoots:=  FindRoots(midPoint, b)
  if numOfLeftRoots =1 then root :=  Bisection(a, midPoint)
  SendInterval(a, midPoint, numOfLeftRoots,root)
  if numOfRightRoots =1 then root := Bisection(midPoint, b)
  SendInterval(midPoint, b, numOfRightRoots, root)
  goto start
```

The above algorithm focuses in obtaining the roots with certainty and robustness. It is designed with high generality in the sense that it can be applied to compute the zeros of any twice continuously differentiable function. Thus, since in many cases the numerical values of various special functions cannot be obtained very accurately and to avoid numerical integration inaccuracies of the user chosen integration method, the above algorithm applies its integration portion at every instance. In cases where the user is certain about the accuracy of the numerical function values in obtaining the total number of zeros, it is evident that the above algorithm can be reconstructed to be more rapid without numerical integrations at every instance.

Although the integral in relation (3.4) is nontrivial, the numerical integration is very fast, since the result — the number of roots in a given interval — is always an integer and thus no high accuracy is required. For example, on an HP-715 computer (with one PA-Risk 7100/75 Mhz processor) the typical elapsed CPU time for the numerical integration is from 9.02 to 32.81 ms, when intervals with length 30–50 are considered. On the average, the typical time required for numerical integration is one third of the time needed for the computation of the isolated zeros, utilizing the modified bisection method (4.1).

6. Experimental results

The algorithm described in Section 5 has been implemented and tested on many problems of various special functions and the results have been quite satisfactory. Our experience is that the algorithm behaves predictably and reliably. With this algorithm and a PVM, one can massively compute *all* the roots of a given twice continuously differentiable function in a given interval, with *certainty*.

The key feature of the proposed algorithm is that it can be straightforwardly implemented in parallel, because the computation of the number of roots using (3.4), the bisection method (4.1), as well as the algorithms of the master and the slaves have large granularity and exhibit low synchronization. This is evident since the parallel version of our algorithm is about ten times faster than the sequential one, when run on a PVM that consists of 10 slave nodes.

To test the performance of the proposed algorithm, we have applied it on the parallel computation of all the roots of special functions of certain orders. To this end, we have tested our approach to a problem where a massive computation of these zeros is required. According to Prof. Á. Elbert, the density property of the zeros of Bessel functions plays an important role and a better insight on the distribution of these zeros is required [2] (see also [4,5], where Joó dealt with oscillation of the circular membranes). To this end, we consider the following set:

$$\mathscr{S} = \bigcup_{n,k=1} \{j_{nk}\} = \bigcup_{j=1} \{x_j\},$$

where j_{nk} is the kth positive zero of the Bessel function of the first kind, $J_n(x)$, and $x_1 < x_2 < \cdots$, and we have tried to extract pieces of information regarding the following product:

$$\mathscr{E}_j = x_j(x_{j+1} - x_j), \tag{6.1}$$

Table 1
Quantitative information for $\mathscr{E}_j$ in Eq. (6.1) for orders $n = 0, 1, \ldots, 10000$ and the number of roots, $\mathscr{N}^r$, within the corresponding exhibited intervals

Interval	Minimum	Mean	Maximum	std.
[10000, 10010]	0.00007450592607	3.99269958773648	47.03825583568963	4.037765
[10000, 10001]	0.00007450592607	3.96253282037896	44.44593478678763	4.172368
[10001, 10002]	0.00111576389384	3.97831989729870	27.68517659088264	3.923294
[10002, 10003]	0.00597613035132	4.02593708771399	31.67826186554454	4.088156
[10003, 10004]	0.00023979790426	3.99404064880568	37.71659298637130	4.025062
[10004, 10005]	0.00080143385603	4.02828978890569	34.70717497269536	4.051591
[10005, 10006]	0.00084866990518	4.01577160014173	44.38887697723416	3.957860
[10006, 10007]	0.00397942852308	3.97103101771882	47.03825583568963	4.107122
[10007, 10008]	0.00318327014514	4.00197484290160	42.26565426253388	4.151968
[10008, 10009]	0.00118115106241	3.97627900894455	32.55679308484231	4.013669
[10009, 10010]	0.00032302494730	3.97416368578276	43.37227316700991	3.883157

$\mathscr{N}^r$: 25059, 2524, 2514, 2482, 2507, 2484, 2490, 2521, 2501, 2517, 2519.

Table 2
Quantitative information for $\mathscr{E}_j$ in Eq. (6.1) for orders $n=0,1,\ldots,2000$ and the number of roots, $\mathscr{N}^r$, within the corresponding exhibited intervals

Interval	Minimum	Mean	Maximum	std.
[0, 2000]	0.00002063847942	4.00165503414487	52.54410751200736	3.962428
[0, 400]	0.00051114207273	4.00816537924939	40.82943057823706	3.881962
[400, 800]	0.00004315974771	4.00268178330445	43.10751779470661	3.961188
[800, 1200]	0.00003193841635	4.00142044530852	40.95811360846550	3.964162
[1200, 1600]	0.00003032675281	4.00068047439606	52.17900499047060	3.963991
[1600, 2000]	0.00002063847942	4.00147908759290	52.54410751200736	3.969533

$\mathscr{N}^r$: 499799, 19963, 59961, 99965, 139976, 179934.

which can be utilized in Elbert's conjecture [2]:

$$\limsup_{j\to\infty} x_j(x_{j+1}-x_j) < \infty.$$

In Tables 1 and 2 we exhibit some quantitative results regarding $\mathscr{E}_j$, for various intervals and orders of $J_n(x)$. We remark that in Table 2, the mean value and the standard deviation of $\mathscr{E}_j$ exhibit, on average, a smoother behavior than in Table 1, due to the larger number of zeros within the considered intervals. We also remark that the mean value and the standard deviation are nearly the same ($\simeq 4$), which may be an interesting point for further investigation.

Moreover, we have computed all the 10 54 942 roots of $J_n(x)$, of order $n=0,1,\ldots,2000$, in the interval $[0,3000]$. These zeros are used to extract information regarding the distances between them. To this end, we have sorted all the computed zeros and we have computed the distances d_i, $i=1,2,\ldots,10\,54\,942$ between all the pairs of consecutive zeros. The mean value of these distances d_i is $\mu(\mathrm{d}_i)=0.00284157510496$, the standard deviation is $\sigma(\mathrm{d}_i)=0.00628066999847$, the minimum distance is $\min_i\{\mathrm{d}_i\}=1.361058821203187\times10^{-9}$ and the maximum distance is $\max_i\{\mathrm{d}_i\}=1.42688040988250$.

7. Conclusion

In this paper, the concept of the topological degree has been utilized to calculate the total number of simple real roots of Bessel functions within a predetermined interval, and to isolate and compute each one of them. For this purpose, we have used the Kronecker's theory and the Picard's extension. Once a zero is isolated, it is computed numerically, utilizing a modified bisection method. This procedure is implemented to run in a parallel computer and is tested using the PVM. The proposed algorithm has large granularity since it consists of large independent portions, and exhibits low synchronization, because no process synchronization is necessary. Furthermore, its performance is fast, robust and predictable.

Of course, our approach can be more efficient and effective utilizing analytic or asymptotic estimates of zeros, in cases of well-known functions where these estimates are available. Also, the algorithm can easily be modified to compute the extrema of a three times continuously differentiable function. Our approach can be used to speed up other similar recently proposed algorithms [6,7,22] with regard to the computation of complex zeros of special functions which is an area of particular

interest [8]. Future work will also include utilization of well-known estimates as well as experiments on various "real life" applications that require fast speed of execution and vast computational resources.

Acknowledgements

We wish to thank Prof. Á. Elbert for many stimulating and useful discussions, constructive comments and valuable suggestions, as well as the anonymous referee, whose remarks helped us to improve the paper.

References

[1] P. Alexandroff, H. Hopf, Topologie, Springer, Berlin, 1935 (reprinted: Chelsea, Bronx, New York, 1965).
[2] Á. Elbert, Some recent results on the zeros of Bessel functions and orthogonal polynomials, this issue, J. Comput. Appl. Math. 133 (2001) 65–83.
[3] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Manchek, V. Sunderam, PVM: Parallel Virtual Machine. A User's Guide and Tutorial for Networked Parallel Computing, MIT Press, Cambridge, 1994.
[4] I. Joó, Exact controllability and oscillation properties of circular membranes, Dissertation for the title doctor of science, Budapest, 1992.
[5] I. Joó, On the control of a circular membrane I, Acta Math. Hungar. 61 (1993) 303–325.
[6] P. Kravanja, O. Ragos, M.N. Vrahatis, F.A. Zafiropoulos, ZEBEC: a mathematical software package for computing simple zeros of Bessel functions of real order and complex argument, Comput. Phys. Commun. 113 (1998) 220–238.
[7] P. Kravanja, M. Van Barel, O. Ragos, M.N. Vrahatis, F.A. Zafiropoulos, ZEAL: a mathematical software package for computing zeros of analytic functions, Comput. Phys. Commun. 124 (2000) 212–232.
[8] D.W. Lozier, Software needs in special functions, J. Comput. Appl. Math. 66 (1996) 345–358.
[9] D.W. Lozier, F.W.J. Olver, Airy and Bessel functions by parallel integration of ODEs, in: R.F. Sincovec, D.E. Keyes, M.R. Leuze, L.R. Petzold, D.A. Reed (Eds.), Proceedings of 6th SIAM Conference on Parallel Processing for Scientific Computing, Vol. 2, SIAM, Philadelphia, 1993, pp. 531–538.
[10] D.W. Lozier, F.W.J. Olver, Numerical evaluation of special functions, in: W. Gautschi (Ed.), Mathematics of Computation 1943–1993: a half-century of computational mathematics, Proceedings of Symposia in Applied Mathematics, Vol. 48, AMS, Providence, RI, 1994, pp. 79–125.
[11] J.M. Ortega, W.C. Rheinbolt, Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, New York, 1970.
[12] E. Picard, Sur le nombre des racines communes à plusieurs équations simultanées, J. Math. Pure Appl. (4 Sér.) 8 (1892) 5–24.
[13] E. Picard, Traité d'analyse, 3rd Edition, Gauthier–Villars, Paris, 1922 (Chapter 4.7).
[14] J. Segura, A. Gil, ELF and GNOME: Two tiny codes to evaluate the real zeros of the Bessel functions of the first kind for real orders, Comput. Phys. Commun. 117 (1999) 250–262.
[15] K. Sikorski, Bisection is optimal, Numer. Math. 40 (1982) 111–117.
[16] T.L. Sterling, J. Salmon, D.J. Becker, D.F. Savarese, How to build a Beowulf: A Guide to Implementation and Application of PC Clusters, MIT Press, Cambridge, 1999.
[17] M.N. Vrahatis, Solving systems of nonlinear equations using the nonzero value of the topological degree, ACM Trans. Math. Software 14 (1988) 312–329.
[18] M.N. Vrahatis, CHABIS: A mathematical software package for locating and evaluating roots of systems of nonlinear equations, ACM Trans. Math. Software 14 (1988) 330–336.
[19] M.N. Vrahatis, A short proof and a generalization of Miranda's existence theorem, Proc. Amer. Math. Soc. 107 (1989) 701–703.
[20] M.N. Vrahatis, T.N. Grapsa, O. Ragos, F.A. Zafiropoulos, On the localization and computation of zeros of Bessel functions, Z. Angew. Math. Mech. 77 (1997) 467–475.

[21] M.N. Vrahatis, O. Ragos, T. Skiniotis, F.A. Zafiropoulos, T.N. Grapsa, RFSFNS: a portable package for the numerical determination of the number and the calculation of roots of Bessel functions, Comput. Phys. Commun. 92 (1995) 252–266.
[22] M.N. Vrahatis, O. Ragos, T. Skiniotis, F.A. Zafiropoulos, T.N. Grapsa, The topological degree theory for the localization and computation of complex zeros of Bessel functions, Numer. Funct. Anal. Optim. 18 (1997) 227–234.
[23] M.N. Vrahatis, O. Ragos, F.A. Zafiropoulos, T.N. Grapsa, Locating and computing zeros of Airy functions, Z. Angew. Math. Mech. 76 (1996) 419–422.